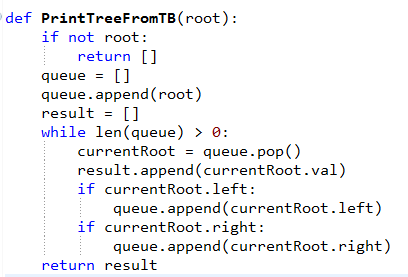
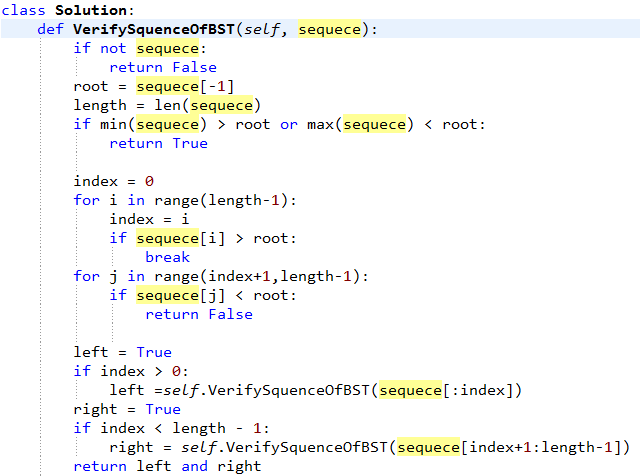
二叉树专题

1、从上到下打印二叉树

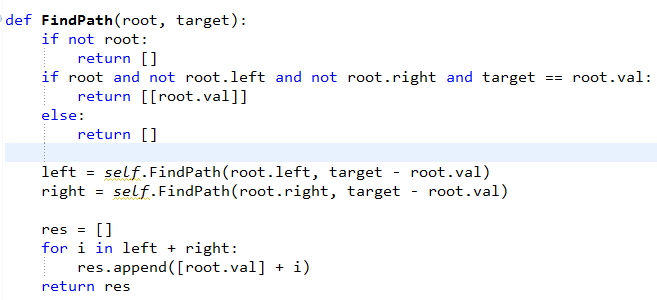


2、判断一个数组是否为某二叉搜索树的后续遍历结果。

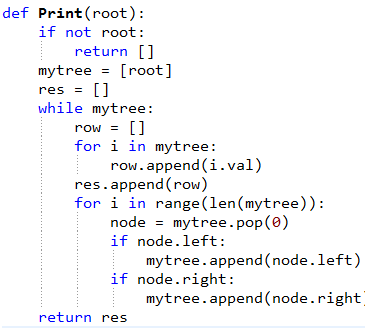
思路：首先最后一个元素是二叉树的根节点，然后左子树都小于根节点，右子树都大于根节点。



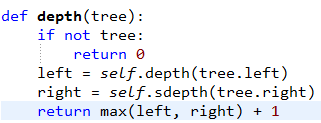
3、二叉树中和为某一值的路径



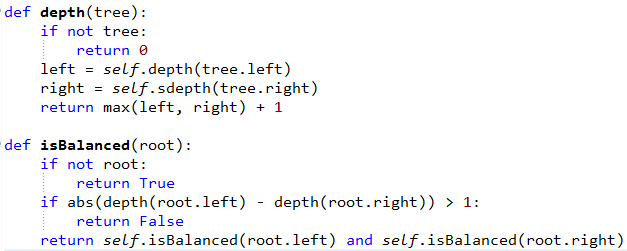
4、多行打印二叉树，要求每一层在一行



5、二叉树的深度

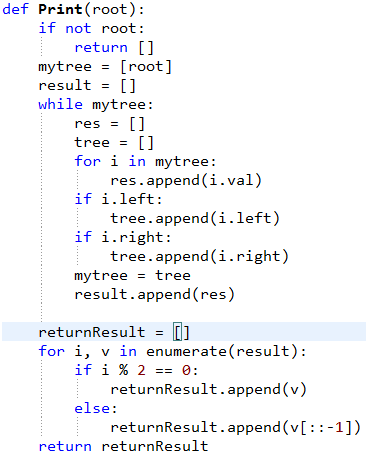


6、判断是否为平衡二叉树



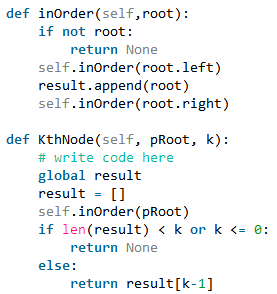
6、之字形打印二叉树

思路：加一个判断条件，奇数层和偶数层



7、二叉搜索树的第k个结点

思路：中序遍历输出一个序列，然后找到序列中第k个数即可。



数组专题

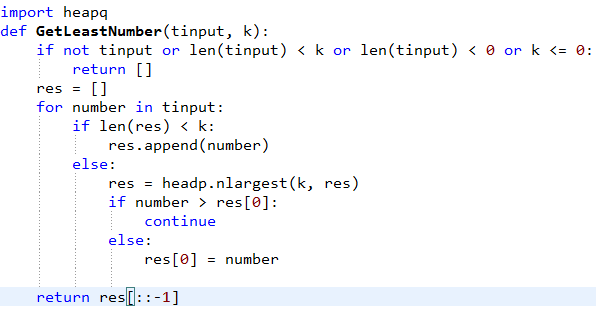
1、数组中出现次数超过一半的数字

(1) 第一种利用python的collections.Counter处理，然后在利用items():

(2) 第二种排序，处理mid应该是该数字，然后遍历对该数字出现次数计数：

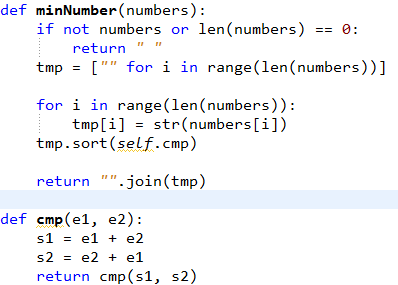
2、最小的k个数

思路：构建最大堆；首先把数组前k个数字构建一个最大堆，然后从第k+1个数字开始遍历数组，如果遍历到的元素小于堆顶的数字，那么久将换两个数字，重新构造堆，继续遍历，最后剩下的堆就是最小的k个数，时间复杂度O(nlog k)。



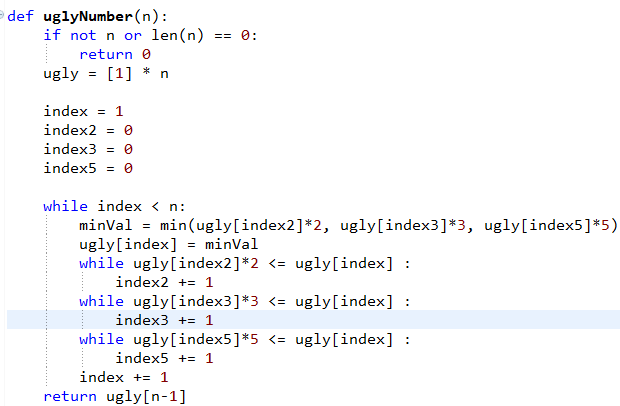
3、把数组排成最小的数

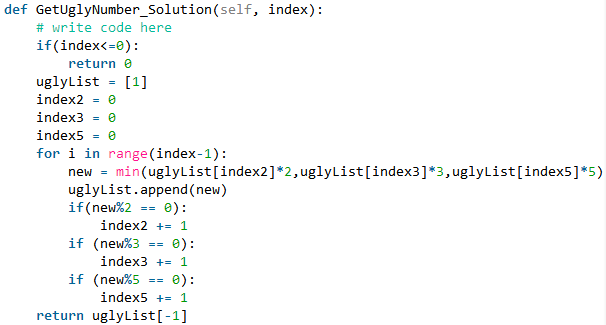
要把数字转成字符形式



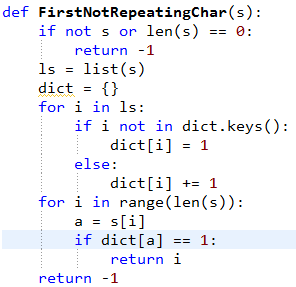
4、丑数

习惯上只包含2、3、5的数称为丑数，例如：6和8都是丑数，习惯上我们把1当做第一个丑数：

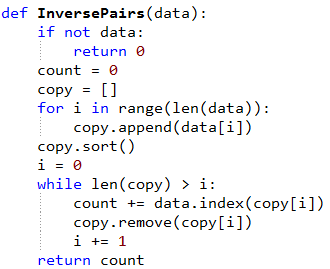




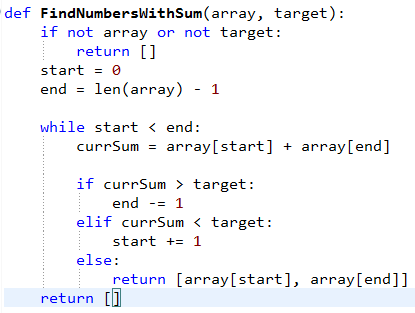
5、第一个只出现一次的字符



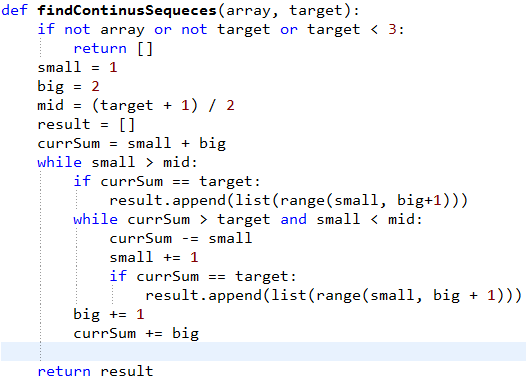
6、逆序数对（未解决）



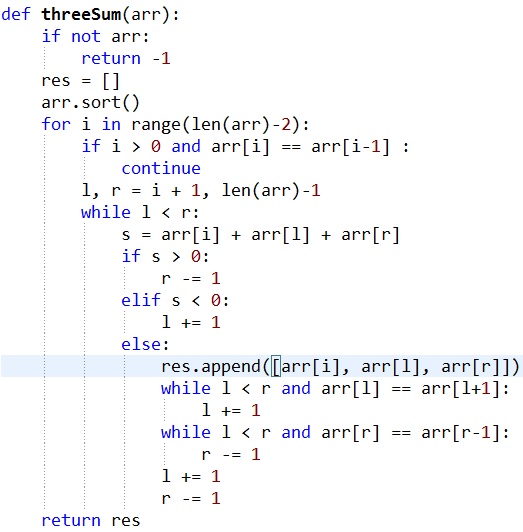
7、和为s的两个数字



8、和为s的连续正数序列



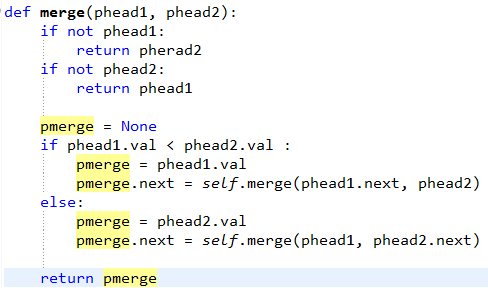
9、和为s的三个数



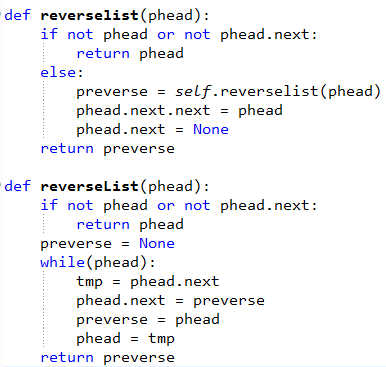
10、4个和为s的数相加

链表专题

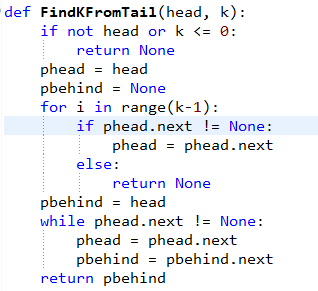
1、合并两个排序的链表



2、反转链表



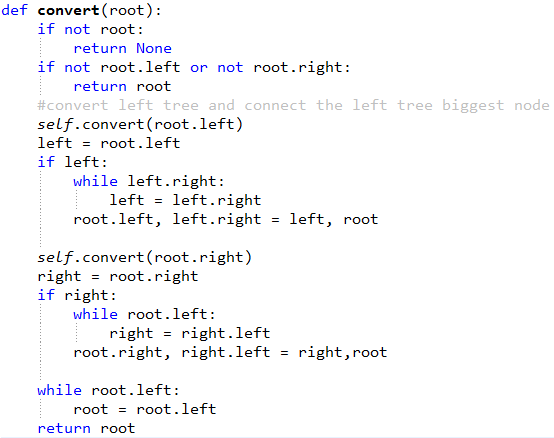
3、链表中的倒数第k个节点



4、二叉搜索树和双向链表

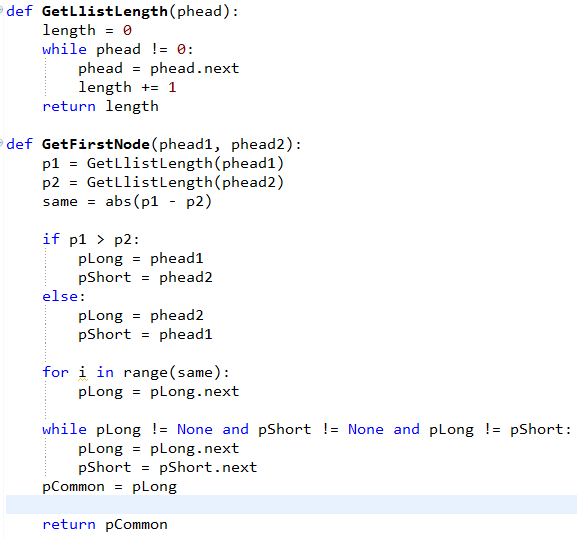
描述：输入一棵二叉搜索树，将该二叉树转换成一个排序的双向链表，要求不创建任何新的节点，只能调整树中指针的指向。

左子树小于根节点，右子树大于根节点，根节点左边连接左子树最大的结点，右边连接右子树最小的结点。



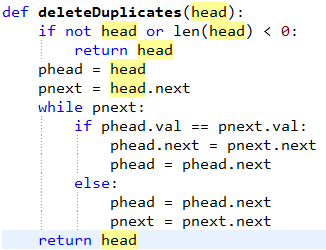
5、两个链表的第一个公共结点

思路：首先依次遍历两个链表，记录两个链表的长度m和n，如果 m > n，那么我们就先让长度为m的链表走m-n个结点，然后两个链表同时遍历，当遍历到相同的结点的时候停止即可。对于 m < n，同理

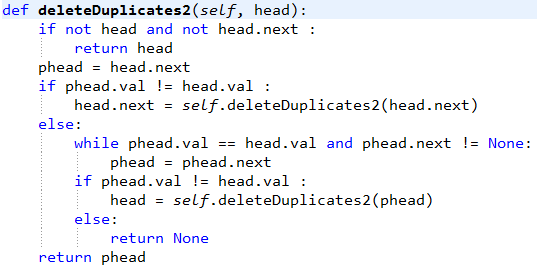


6、删除链表中的重复结点（保留重复结点）

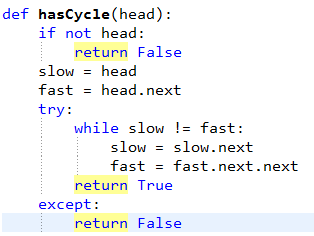
思路：1->1->2；变成1->2



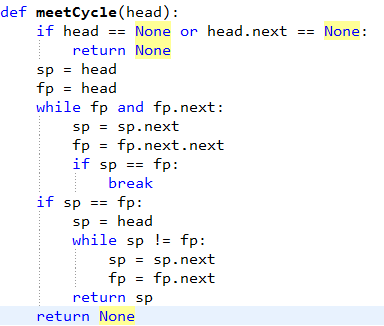
7、删除链表中的重复结点（不保留重复结点）



8、判断链表是否存在环

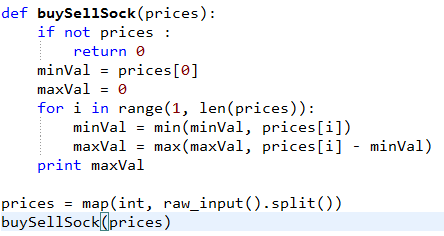


9、判断链表环入口（快慢指针实现）



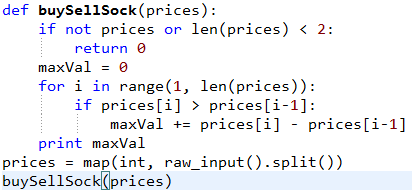
**动态规划**

1、最好的时间买入和卖出股票（只可以交易一次）



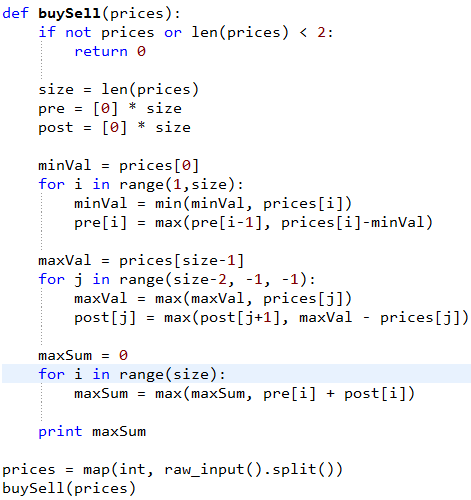
2、买入卖出股票的最大利润（允许多次交易）

思路：判断前一个比后一个小，然后相减依次迭加就好了！！！

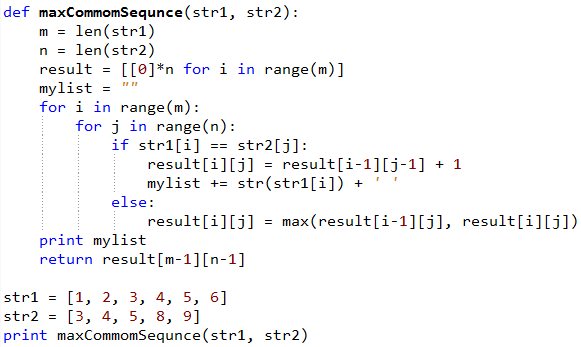


3、买入卖出股票的最大利润（允许两次交易）

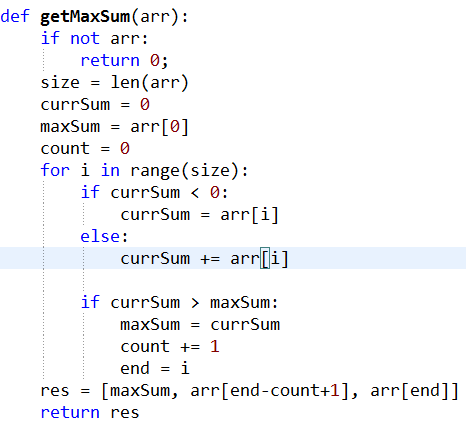
思路：找到一个结点i， 判断i之前的最大利润和i之后的最大利润。



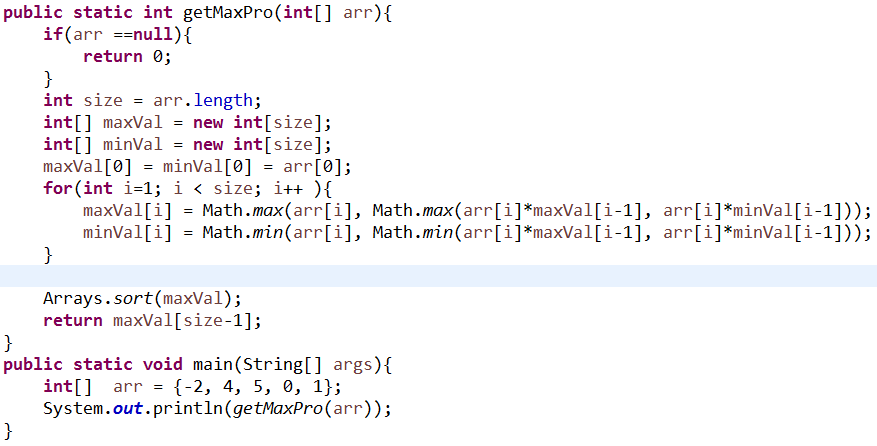
4、最长公共子序列

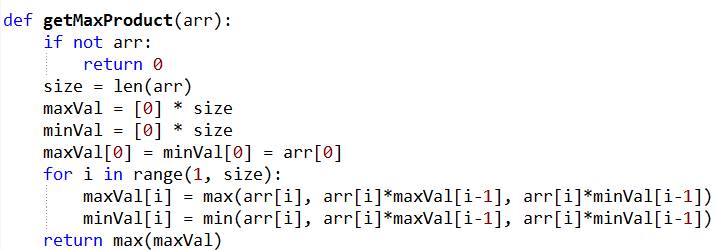
思路：计算两个序列的长度，利用动态规划的思想，新建一个m\*n的数组，加入相等的话，dp[i][j]=dp[i-1][j-1]+1；否则dp[i][j]=max(dp[i-1][j],dp[i[j-1])

5、最大子序列和

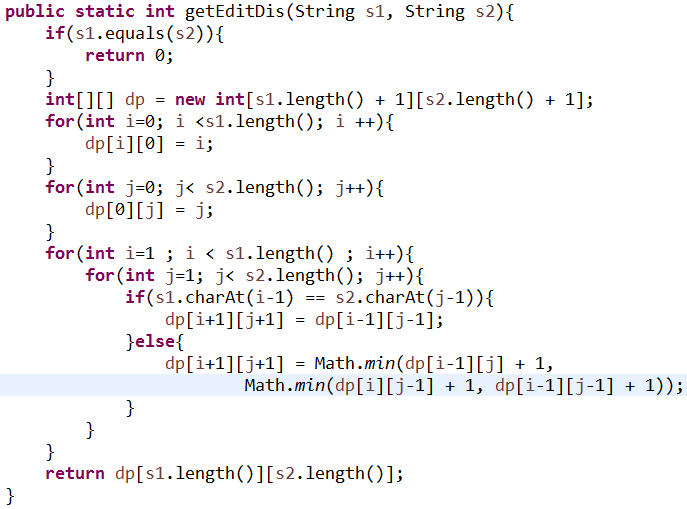


6、最大子序列乘积

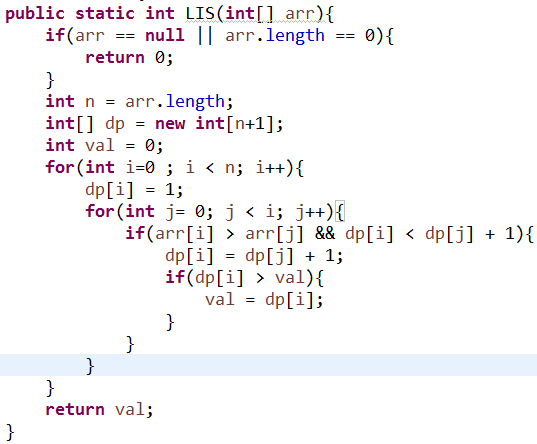




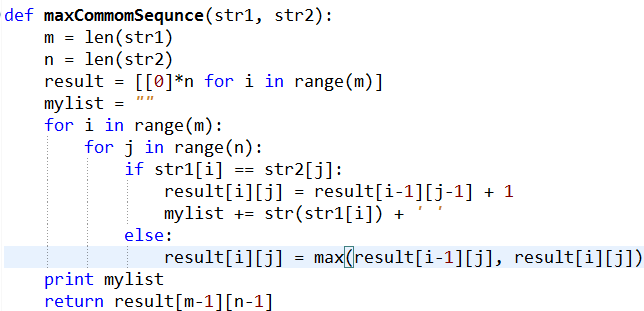
7、最小编辑距离



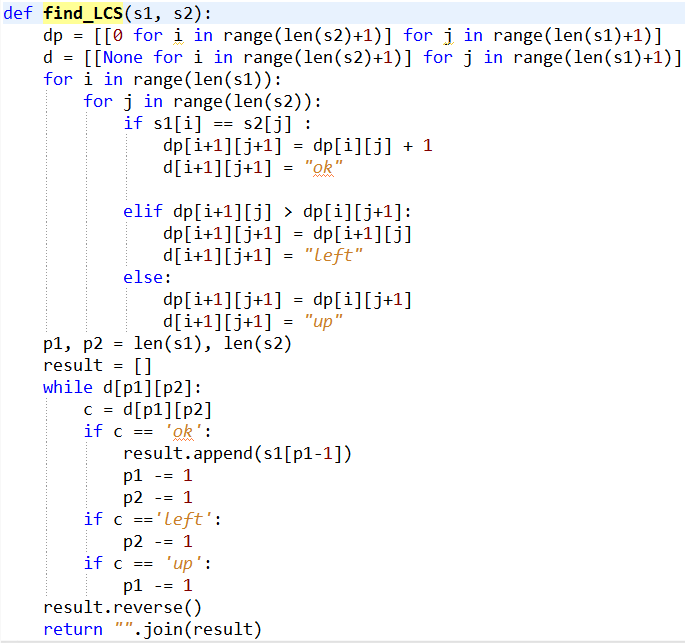
8、最长递增子序列



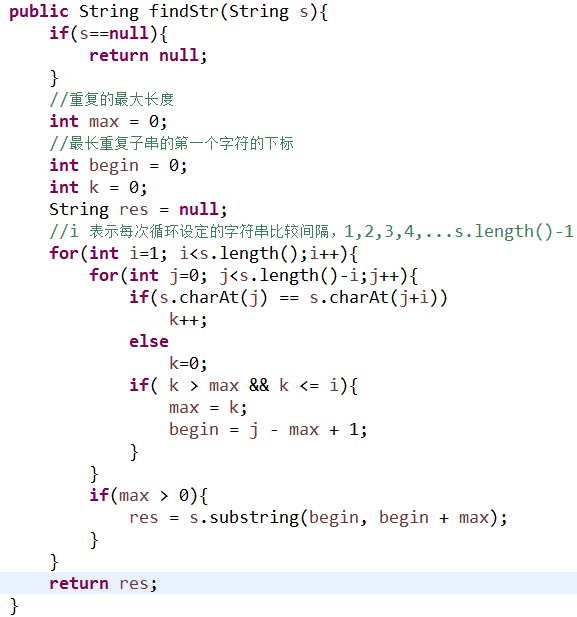
9、最长公共子串



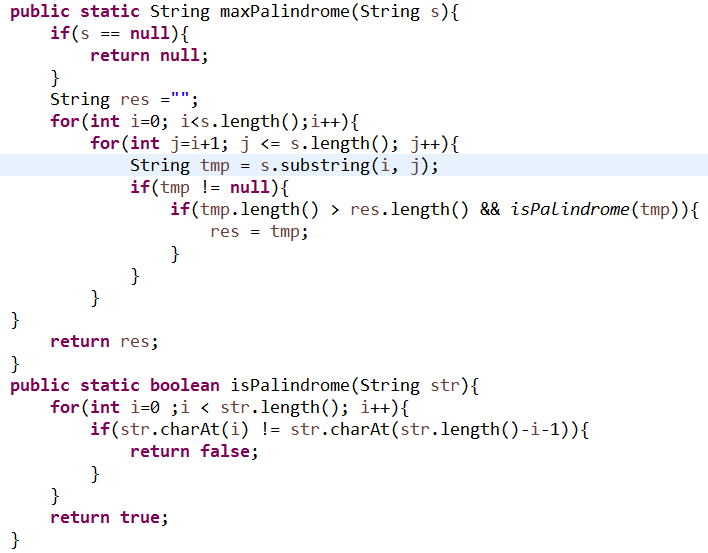
10、最长公共子序列



11、最长不重复子串



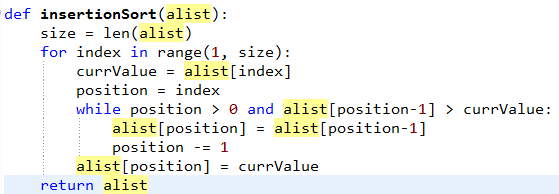
12、最长回文子串



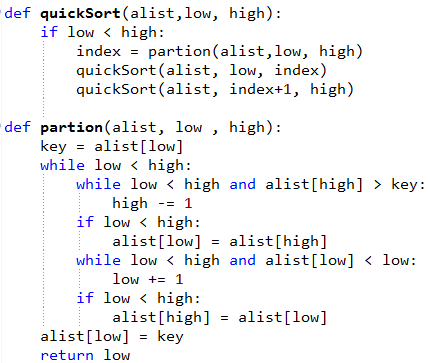
**数据结构**

1、插入排序

选择序列中的第一个元素作为有序序列，逐渐将后面的元素插入到前面的有序序列中。



2、快速排序



3、